CSE142 Sample Final Exam
Autumn 2019

1. Reference Mystery, 7 points. The following program produces 3 lines of output. Write the output in the box below, **exactly** as it would appear on the console.

```
import java.util.*;
public class Point {
   int x;
   int y;

   public Point(int initX, int initY) {
      x = initX;
      y = initY;
   }
}

public class ReferenceMystery {
    public static void main(String[] args) {
        Point p = new Point(11, 22);
        int[] a = {33, 44};
        int n = 55;

        mystery1(p, a, n);
        System.out.println(p.x + "," + p.y + " " + Arrays.toString(a) + " " + n);

        a[0] = a[1];
        p.x = p.y;

        n = mystery2(a, n);
        System.out.println(p.x + "," + p.y + " " + n);
    }

    public static int mystery1(Point p, int[] a, int n) {
        n = 0;
        a[0] = a[0] + 11;
        a[1] = 77;
        p.x = p.x + 33;
        System.out.println(p.x + "," + p.y + " " + Arrays.toString(a) + " " + n);
        return n;
    }

    public static int mystery2(int[] a, int n) {
        n = a[0];
        a[0] = a[0] + 11;
        a[1] = n + 11;
        return n;
    }
}
```

2. Array Simulation, 10 points. You are to simulate the execution of a method that manipulates an array of integers. Consider the following method:

```
public static void mystery(int[] a) {
    for (int i = 1; i < a.length - 1; i++) {
        a[i] = (a[i - 1] + a[i + 1]) / 2;
    }
}
```

In the left-hand column below are specific arrays of integers. You are to indicate in the right-hand column what values would be stored in the array after method mystery executes if the integer array in the left-hand column is passed as a parameter to mystery.

Original Array	Final Array
[1, 1, 3]	_____
[2, 1, 2, 4]	_____
[6, 13, 0, 3, 7]	_____
[-1, 6, 3, 5, -3]	_____
[7, 2, 3, 1, -3, 12]	_____

3. Inheritance Mystery, 8 points. Consider the following classes:

```
public class Seal extends Bluth {
    public String toString() {
        return "Seal " + super.toString();
    }
}

public class Buster extends Bluth {
    public void method2() {
        System.out.println("Buster 2");
    }

    public String toString()() {
        return "Buster";
    }
}

public class Bluth {
    public void method1() {
        method2();
        System.out.println("Bluth 1");
    }

    public void method2() {
        System.out.println("Bluth 2");
    }

    public String toString() {
        return "Bluth";
    }
}

public class Gob extends Bluth {
    public void method1() {
        System.out.println("Gob 1");
    }

    public void method2() {
        System.out.println("Gob 2");
    }
}
```

Given the classes to the left, write the output produced by the client code below **exactly** as it would appear on the console.

```
// client code
public static void main(String[] args) {
    Bluth[] bluths = { new Seal(), new Gob(), new Bluth(), new Buster() };
    for (int i = 0; i < bluths.length; i++) {
        System.out.println(bluths[i]);
        bluths[i].method1();
        bluths[i].method2();
        System.out.println();
    }
}
```

4. ArrayList Debugging, 5 points. Consider a static method called cloneOddsRemoveEvens that takes an ArrayList of integer values as a parameter and that replaces each odd number with two of that number and that removes all even values. For example, suppose that a variable called list stores the following sequence of values:

```
[3, 8, 19, 42, 7, 26, 27, -8, 193, 204, 6, -4]
```

and we make the following call:

```
cloneOddsRemoveEvens(list);
```

Afterwards the list should store the following sequence of values:

```
[3, 3, 19, 19, 7, 7, 27, 27, 193, 193]
```

Notice that each odd number has been duplicated (3, 19, 7, 27, 193) and that the even values have been removed (8, 42, 26, -8, 204, 6, -4). Below are several examples of how an ArrayList should be changed by the method.

ArrayList contents before call	ArrayList contents after call
[]	[]
[82]	[]
[3, 8, 15]	[3, 3, 15, 15]
[2, 4, 6, 8]	[]
[17, 5, 3, 9]	[17, 17, 5, 5, 3, 3, 9, 9]
[1, 2, 3, 4, 5, 6]	[1, 1, 3, 3, 5, 5]

The following is a proposed implementation of cloneOddsRemoveEvens:

```
public static void cloneOddsRemoveEvens(ArrayList<Integer> list) {
   for (int i = 0; i < list.size(); i++) {
      int val = list.get(i);
      if (val % 2 == 0) {
         list.remove(i);
      } else {
         list.add(i + 1, val);
      }
   }
}
```

This implementation has one or more bugs. Modify the implementation so that it behaves as described above. Your modified method should retain the same basic approach to the problem as the buggy implementation; you should not write an entirely new implementation.

You may not construct any extra data structures to solve this problem. You must solve it by manipulating the ArrayList you are passed as a parameter. See the cheat sheet for a list of available ArrayList methods.

5. File Processing, 10 points. Write a static method called underline that accepts as a parameter a Scanner containing an input file and that prints to System.out the same text with certain lines underlined. The lines to be underlined all begin with a period. The period should not be printed. You should print the text that follows the period on a line by itself followed by a line of dashes equal in length to the text that follows the period. For example, consider the following input:

```
.Statement of Purpose
I didn't expect to major in computer science until I took cse142.
I liked it more than I expected and that got me hooked on cs.

.High School Performance
I got very good grades in high school, graduating in the top 10% of
my class.

.College Performance
I have done well in my college classes, with an overall gpa of 3.5.
```

If the text above is stored in a Scanner called input and we make this call:

```
underline(input);
```

the method should print the following output to System.out:

```
Statement of Purpose
--------------------
I didn't expect to major in computer science until I took cse142.
I liked it more than I expected and that got me hooked on cs.

High School Performance
-----------------------
I got very good grades in high school, graduating in the top 10% of
my class.

College Performance
-------------------
I have done well in my college classes, with an overall gpa of 3.5.
```

Notice that some of the input lines can be blank lines. You may not construct any extra data structures to solve this problem.

6. File Processing, 10 points. Write a static method called redact that takes as a parameter a Scanner containing a text with special markers indicating sensitive words that are to be replaced. It prints the resulting text with each word on a separate line. The idea is that the text contains potentially classified information and extra "content markers" have been included throughout to indicate classified material. The special string "CLASSIFIED" appears in various parts of the text to indicate sensitive material (with exactly this casing). Each occurrence of the string will be followed by an integer indicating how many words are to be redacted. For each of those words, you should print the text "[redacted]" instead of printing the word. For example, if a Scanner called text contains the following:

```
four score CLASSIFIED 3 and seven years ago our CLASSIFIED 1 fathers
brought forth CLASSIFIED 2 on this continent
```

There are three indications of classified material. If you were to call the method as follows:

```
redact(text);
```

the following output should be produced:

```
four
score
[redacted]
[redacted]
[redacted]
ago
our
[redacted]
brought
forth
[redacted]
[redacted]
continent
```

You are to exactly reproduce the format of this output. Notice that line breaks in the input are not meaningful. You may not construct any extra data structures to solve this problem.

7. Arrays, 10 points. Write a static method called minGap that takes an array of integers as a parameter and returns the minimum gap between adjacent values in the array. The gap between two adjacent values in a list is defined as the second value minus the first value. Consider a variable called list defined as follows:

```
int[] list = {1, 3, 6, 7, 12};
```

The first gap is 2 (3 - 1), the second gap is 3 (6 - 3), the third gap is 1 (7 - 6) and the fourth gap is 5 (12 - 7). Thus, the call:

```
minGap(list)
```

should return 1 because that is the smallest gap in the list. Notice that the minimum gap could be a negative number. If the list has fewer than 2 elements, your method should return 0. Below are several examples of what value would be returned for a given array.

Array passed as parameter	Value Returned
{}	0
{8}	0
{4, 17}	13
{3, 5, 11, 4, 8}	-7
{2, 8, 8, 10, 15}	0

You may not construct any extra data structures to solve this problem and your method should not alter the array passed as a parameter.

8. Critters, 15 points. Write a class called Orca that extends the Critter class. The instances of the Orca class are always black. Each Orca is always either in moving-mode or in turning-mode. They start out in moving-mode. While in moving-mode, they try to hop forward if possible until they have hopped four times, at which point they switch into turning-mode. If it is not possible to hop while in moving-mode, an Orca instead infects whatever is in front of it. When in turning-mode, the Orca turns left twice and then switches back to moving-mode. Don't worry about the fact that if the Orca encounters a wall while in moving-mode, it gets stuck trying to infect the wall indefinitely. The Orca displays itself as "M" while in moving-mode and as "T" while in turning-mode.

9. Arrays, 15 points. Write a static method called expand that takes an array of count/value pairs and that constructs and returns a new array containing the expansion of those count/value pairs. For example, suppose that a variable list has been defined as follows (pairs indicated for emphasis):

```
int[] list = {3, 8, 4, 2, 0, 42, 5, 1};
              |  |  |  |  |   |  |  |
              +--+  +--+  +---+  +--+
              pair  pair  pair   pair
```

This array indicates that the result should have 3 occurrences of 8 followed by 4 occurrences of 2 followed by 0 occurrences of 42 followed by 5 occurrences of 1. So, the call expand(list) would construct and return the following array:

```
{8, 8, 8, 2, 2, 2, 2, 1, 1, 1, 1, 1}
```

Solving this problem will require making two passes through the original array. In the first pass, your method should compute the overall length required for the new array. In the second pass, it should fill up the new array to be returned. You may assume that the array passed to your method has a legal sequence of count/value pairs (so it has even length) and has no negative counts. Notice, however, that a count might be 0, as in the example above.

You may not construct any String objects or extra data structures other than the array that is returned to solve this problem. You may not modify the array that is passed in.

10. Programming, 10 points. Write a static method called findIndexes that takes an integer n and an array of integer values as parameters and finds the indexes of the first occurrence of the values 0 through n-1 inclusive, returning those indexes in an array. If some value is not found, then the value at the corresponding index should be -1. For example, suppose that a variable called list has been defined as follows:

```
int[] list = {3, 8, 5, 7, 12, -8, 3, 1, 4, 0, 6};
```

Consider the call of findIndexes(10, list). Because the first parameter of the call is 10, the method will be searching for the index of the first occurrence of each of the values between 0 and 9 inclusive. The resulting array that is constructed and returned will be of length 10. In the input array, the value 0 first appears at index 9. The value 1 first appears at index 7. The value 2 does not appear at all. The value 3 first appears at index 0. And so on. Therefore, the array that is constructed and returned will be:

```
{9, 7, -1, 0, 8, 2, 10, 3, 1, -1}
```

Note that because the value 2 did not occur in list, the value at index 2 in the resulting array is -1.

If the variable list had instead been defined as follows:

```
int[] list = {2, 7, 9, 3, 4, 5, 2, 0, 6, 42};
```

then a call of findIndexes(6, list) would return this array:

```
{7, -1, 0, 3, 4, 5}
```

As in the examples above, you cannot make any assumptions about the range of values stored in the array to be examined, but you may assume that the parameter n is greater than 0. You may not construct any String objects or extra data structures other than the array that is returned to solve this problem.